VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Uživatelská rozhraní aplikací - User's interfaces of applications

Bakalářská práce

2009 Lenka Skanderová

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

V Ostravě 7. května 2008 .....

Ráda bych na tomto místě poděkovala všem, kteří mi s prací pomáhali, neboť bez nich by tato práce nevznikla.

Abstrakt

Obsahem práce na téma Uživatelská rozhraní aplikací je především shrnutí všeobecně platných pravidel pro jejich tvorbu. Zabývám se v ní zvyklostmi uživatelů, zkoumáním nejrůznějších uživatelských rozhraní, jež uživatelé běžně používají v každodenním životě, a jejich zhodnocením. Dále jsem měla za úkol vybrat si uživatelské rozhraní, jež je v zásadním rozporu se zmíněnými pravidly a zásadami, a navrhnout prototyp dle těchto zásad. Jako námět k přepracování jsem si zvolila program s názvem Dia, jež slouží k vytváření diagramů, ať už jde o diagramy ER nebo diagramy vývojové. Tento program byl vhodný zejména díky užívání diskutabilních postupů, jako například dvě oddělená okna, jež zabírají místo v liště apod. Pokusila jsem se tedy navrhnout aplikaci, jež by měla mít stejné funkce jako aplikace Dia a dle zásad tvorby uživatelského rozhraní se zde nebudou vyskytovat stejné typy chyb.

Klíčová slova: uživatelská rozhraní, aplikace, prototyp

Abstract

The content of my work on the topic User's interface of application is above all summary of generally valid rules for their creation. I am engaged in user's convention, investigation of assorted user's interfaces generally used in their common life, and their estimation. Further I should choose some user's interface, which is in without complying with mentioned rules and principles, and project prototype by this principles. As the subject for remake I have choosen the program named Dia, which makes for production of diagrams, for example ER diagrams or development diagrams. This program fir especially thanks by using of controversial processes, for example two separated windows, which occupy place in start bar etc. I have tried to project an application, which should have the same functions as application Dia and by the principles of creation of user's interface the same type of mistakes will not abound here.

Keywords: user's interface, application, prototype

Obsah

Seznam obrázků

1 Úvod

V první části své práce jsem vytvořila jednoduchou příručku, která stručně popisuje, jak se uživatelská rozhraní aplikací tvoří. Jde o jednoduché zpracování základních zásad, které by měly zaručit úspěšný výsledek jak programátorům, tak především uživatelům, jež budou s programem pracovat. Je třeba si uvědomit, že uživatelé nejsou vševědoucí a mnozí z nich mají k počítačům a programům naprostou averzi. Proto se zde objevuje problém uživatelského rozhraní, které by mělo vše zjednodušovat, být k uživatelům přívětivé a ukazovat jim cestu – jak správně pracovat. Jednou kdosi řekl, že v jednoduchosti je krása. A když se nad tím zamyslíme, uvidíme, že to byl moudrý člověk, který věděl, že jednoduchost a zjednodušení problému ušetří spousty času, práce, ale také lidských nervů. Pojďme se proto pokusit řídit se těmi pravidly, která jsou zatím v oblasti uživatelského rozhraní definována.

2 Nejčastější chyby uživatelského rozhraní

V uživatelských rozhraních se vyskytuje celá řada problémů a zapeklitých otázek. My se zde budeme věnovat těm nejobvyklejším chybám, jež nalezneme jak v prostředí Internetových aplikací, tak například v běžných desktopových aplikacích, které využíváme ve svém volném čase:

1. Nepřehlednost
2. Nepatřičný vzhled
3. Nadbytečné grafické prvky

2.1 Nepřehlednost

Většina chyb, kterých se při návrhu grafického uživatelského rozhraní dopouštíme, vychází z chybného návrhu tzv. mentálního modelu uživatele. Co přesně znamená tento pojem? Mentální model lze vysvětlit následovně: Každý člověk se snaží nalézt podvědomě ve věcech řád, tzn. snaží se vytvořit si mentální model.

Pokud se mentální model podaří nalézt, dosáhneme u uživatele:

1. Pocit jistoty, uživatel nabývá v této oblasti své práce sebevědomí a přestává se práce děsit.
2. Pocit důvěry v tento program, která pramení z důvěry k sobě samotnému. Uživatel získá pocit, že je schopen vyřešit i nové situace, které se v tomto programu či aplikaci objeví.
3. Pocit víry, že program danou nepředvídanou situaci bude umět řešit.

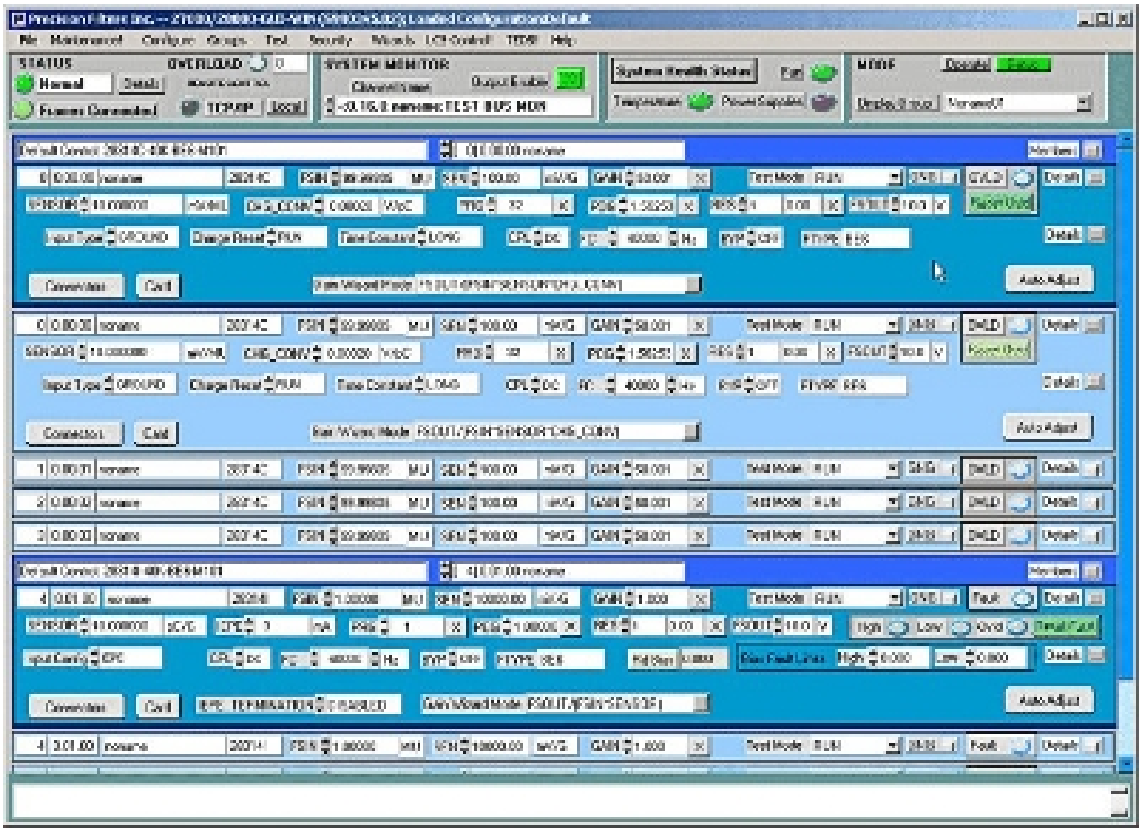
Jakmile se mentální model nalézt nepodaří, nastupuje u uživatele:

1. Pocit nejistoty, uživatel s aplikací nerad pracuje, nevěří jí ani sobě.
2. Stres, který u mnoha lidí přerůstá v deprese.
3. Frustrace

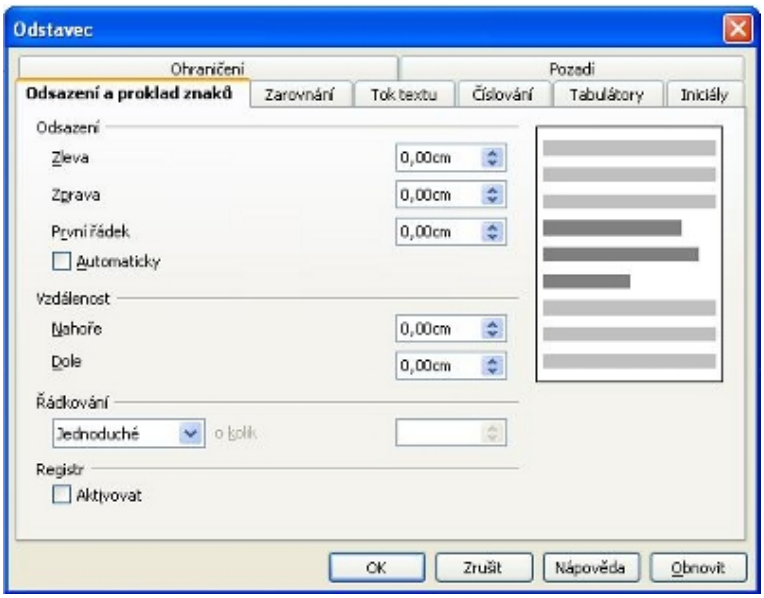
2.1.1 Srovnání

Abychom viděli názorně, co přesně znamená nepřehlednost, srovnejme si dva příklady, z nichž jeden bude reprezentovat přehlednost a druhý nepřehlednost.

1. Absolutní chaos reprezentuje program firmy Precision filtres, inc s názvem 28314 Quad Dynamic Charge viz. obrázek 1.



Obrázek 1: Aplikace firmy Precision filtres



Obrázek 2: Aplikace OpenOffice Writer

2. Naproti tomu správné rozvržení a naprostou přehlednost demonstruje program OpenOffice Writer, konkrétně pro ukázku je vybrán formulář pro upravení odstavce viz. obrázek 2.

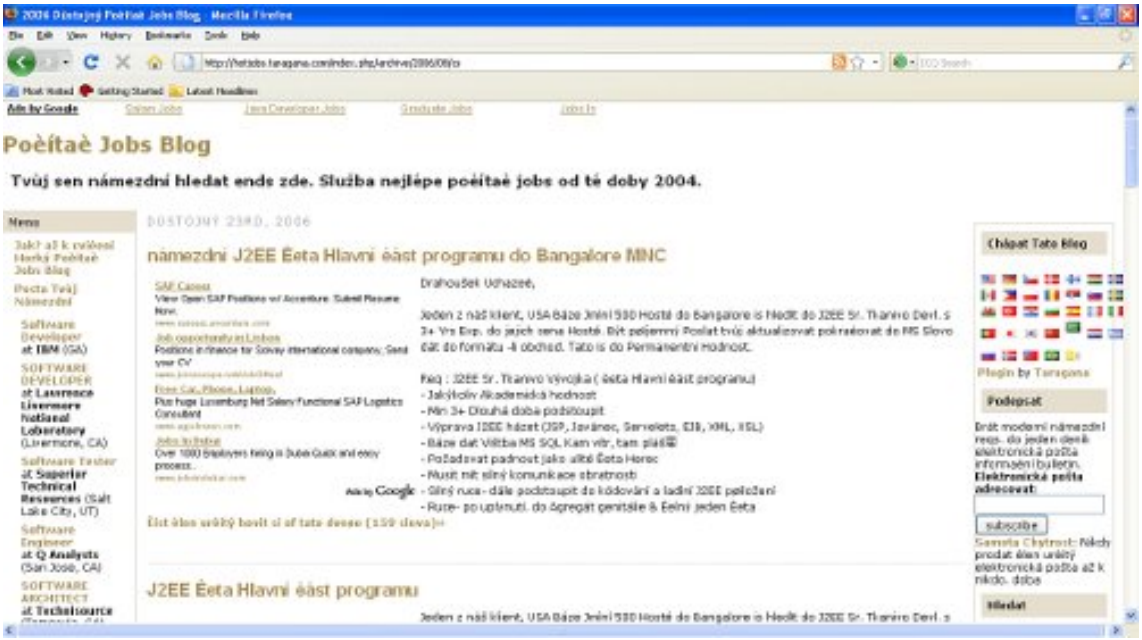
V prvním případě jsou veškeré komponenty neuspořádány, uživatel si nemůže vytvořit mentální model. Komponent je navíc v uživatelském formuláři příliš mnoho, takže už pouhý první pohled uživatele od práce s tímto programem odrazuje. Komponenty postrádají jakýkoliv řád, čímž se dostáváme k dalšímu základnímu problému grafického uživatelského rozhraní, a to je nepatřičný vzhled.

V druhém případě, tedy u programu OpenOffice Writer,mají komponenty řád, jsou zarovnány a řadí se do ucelených rámců, které uživateli pomohou v orientaci a ušetří tak čas. Optimální počet komponent dává uživateli pocit jednoduchosti a dodává mu tak sebevědomí.

Čtenář může namítnout, že zde uvádím pouze programy a že ve webových aplikacích se nic podobného stát nemůže. To by však byl velký omyl – domnívat se, že všechny webové stránky oplývají harmonickým uspořádáním. Abychom tedy plně pochopili, co pod pojmem *Nepřehlednost* rozumíme, uvedu zde ještě jeden příklad právě z oblasti Internetu.

1. Podívejme se na blog s názvem Počítač Jobs Blog, kde nepřehlednost přímo bije do očí.

2. Naproti tomu zhodnoťte stránku prestižní firmy L'OREAL



Obrázek 3: Webová stránka - Computer Jobs Blog

Autor blogu si zjevně s návrhem velkou práci nedal. Objevuje se zde sice náznak levého sloupce, středu a pravého sloupce, ale pro rychlou orientaci s tímto skutečně nevystačíme. Navíc při trochu bližším prozkoumání zjistíme, že o češtině zde nemůže být ani řeč, takže návštěvník této stránky bude pravděpodobně raději hledat jinou alternativu, jak najít právě to, co potřebuje. O nepřehlednosti vás přesvědčí obrázek 3.

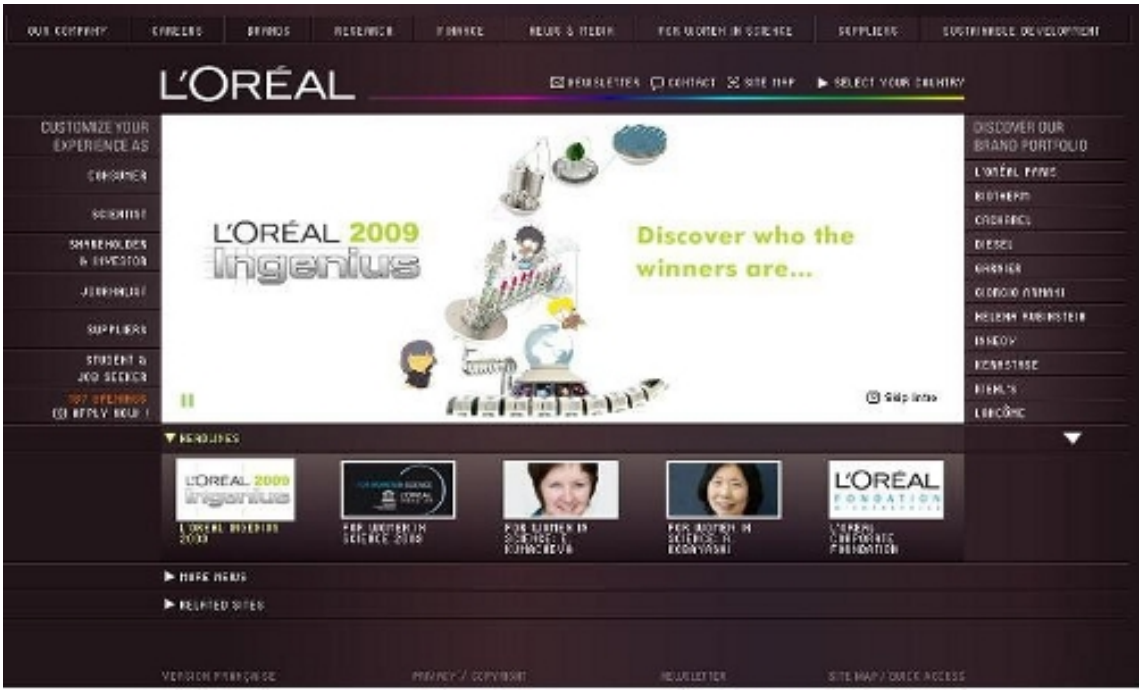
Stránka firmy L'OREAL je velmi stylová. Menu je rozmístno kolem animace, jež poutá hlavní pozornost. O tom však až později. Nyní nás zajímá především přehlednost, což je zde splněno. Menu je členěno na dvě části. První se nachází nahoře, což dává dojem nadřazenosti. Kolem obrázku jsou pak uspořádány odkazy, v nichž se lze snadno orientovat a uživatel či uživatelka rychle nalezne to, co hledá viz obrázek 4.

2.1.2 Osm zlatých pravidel pro tvorbu uživatelského rozhraní

1. Konzistence. Konzistentní terminologie, směřování k tvorbě stereotypů.

2. Respektování široké skupiny uživatelů.

3. Zpětná vazba - uživatel potřebuje mít informace o tom, zda akce proběhla či nikoliv. V tomto případě rozlišujeme silnou a slabou zpětnou vazbu. Silná zpětná vazba se objevuje v těch momentech, kdy má informace pro uživatele zásadní význam. Slabá zpětná vazba se považuje spíše za doplňující.



Obrázek 4: Webová stránka firmy L'OREAL

4. Navigace uživatele - rozdělujte posloupnost po sobě jdoucích kroků do logických na sebe navazujících kroků v GUI, které respektují pracovní postup.

5. Předcházení chybám - nedovolte uživatelům dělat chyby, jimž je možno předcházet, tzn. tlačítka, k nimž nemá mít uživatel v dané chvíli přístup, nechť nejsou aktivní. V případě, že uživatel chybu udělá, je nutno jej o této chybě informovat srozumitelnými hláškami, tzn. jazykem, jemuž rozumí uživatel.

6. Možnost vrátit se zpět a tolerance k chybám

7. Předvídatelné uživatelské rozhraní - uživatel ovládá aplikaci, nikoliv aplikace uživatele. Uživatel musí mít pocit, že má nad aplikací kontrolu.

8. Nepřetěžování krátkodobé paměti uživatele, přehlednost.

Nenuťte uživatele, aby si postupy v aplikaci pamatoval. Zde je důležitá přehlednost.

2.1.3 Uspořádání grafických prvků v aplikaci

Co je potřeba vzít v úvahu:

1. Uživatel prohlíží obsah aplikace od levého horního rohu po směru hodinových ručiček.

2. V našich zeměpisných podmínkách se čte zleva - doprava a shora – dolů.

3. Rozmístění a posloupnost ovládacích prvků musí respektovat tok informací.

Zásady uspořádání grafických prvků v aplikaci

1. Zásada vyváženosti - uživatelské okno musí být rovnoměrně zaplněno, a to jak horizontálně, tak vertikálně.

2. Zásada souměrnosti - souměrné uspořádání prvků by mělo být stejné na levé i pravé straně okna.

3. Zásada pravidelnosti - rozměry stejných ovládacích prvků, jejich tvar, barva vzdálenosti mezi nimi, by měly být všude tam, kde je to vhodné, jednotné.

4. Zásada předvídatelnosti - uspořádání všech prvků by mělo být stejné nebo aspoň velmi podobné napříč všemi okny aplikace.

5. Zásada následnosti - ovládací prvky by měly být rozmístěny logicky podle své důležitosti a významu.

6. Zásada účelnosti

7. Zásada jednotnosti

8. Zásada proporce - klademe důraz na soulad mezi prvky, na jejich rozměry a především na celek.

9. Zásada jednoduchosti - okno by mělo obsahovat přiměřený počet ovládacích prvků.

10. Zásada seskupování - prvky seskupujeme podle významu a účelu.

2.2 Nepatřičný vzhled

Do nepatřičného vzhledu můžeme řadit špatné zvolení barev, jejichž kontrast na uživatele působí naprosto nevyváženě a rušivě, nevhodně umístěné animace, špatná navigace. Zkrátka všechno, co uživateli brání se soustředit, popřípadě co odvádí jeho pozornost jinam, než je potřeba.

Na co se soustředit:

1. Organizace textu

2. Použití stylů písma

3. Volba pozadí a používání barev

2.2.1 Organizace textu

Pokud reprezentujeme nějaký rozsáhlejší text, je třeba jej rozdělit do sekcí, zvýraznit nadpisy a dát tak textu řád. Opět zde platí, že mentální model musí být uživatelem vytvořen správně. U webových aplikací, kde převážně reprezentujeme delší texty, by měl být mentální model velmi podobný s mentálním modelem knihy.

Zásady organizace textu:

1. Uživatel by si měl co nejrychleji vytvořit mentální model, který mu poskytne přesné informace, o čem stránka či aplikace je.

2. Logické členění textů na myšlenky, schémata, příklady atd.

3. Nejdůležitější informace by měly být reprezentovány nadpisy, případně obrázky.

2.2.1.1 Srovnání

1. Stránky s životopisem Ludwiga van Beethovena reprezentují špatný příklad.

2. Naproti tomu stránky první pomoci mají písmo členěno správně.

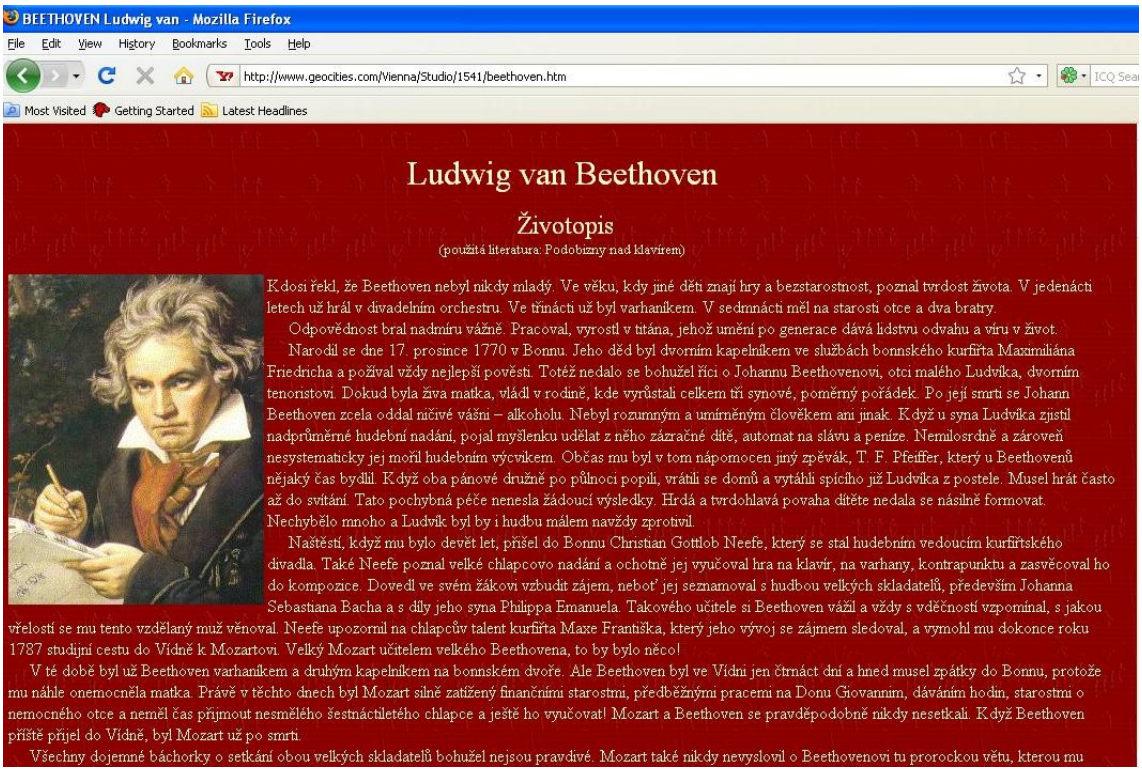
Z obrázku 5 je na první pohled jasné, že čtenář u textu příliš dlouho nevydrží. Text je souvislý, odstavce jsou v něm velmi málo patrné. Na druhé straně je zde dobře zvolen font patkový, jelikož ze jedná o dlouhý text.

Stránky první pomoci viz. obrázek 6 mají svá mínus, ale členění textu zde mají velmi dobré. Všimněte si, že text není příliš dlouhý, je dělen do přehledných oddílů, jejichž nadpis je zvýrazněn.

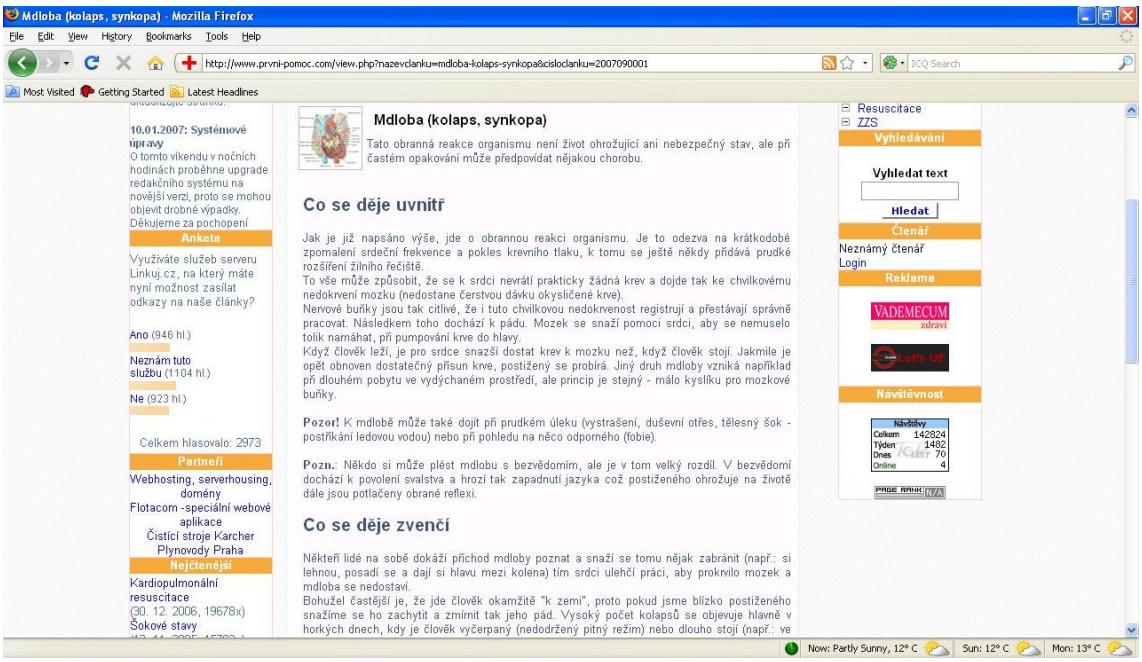
2.2.2 Použití stylů písma

V knihách obvykle vídáme texty psány patkovým písmem, protože patkové písmo vede oko po řádku. V aplikacích, zejména u webových aplikací využíváme zvláštní fonty, jež patku nemají, zato znaky mají větší tloušťku a jsou dále od sebe (např. Arial, Verdana). Říkáme jim bezpatková a jsou uzpůsobena pro obrazovky počítačů. Pro lepší představu se podívejme na obrázek 7.

2.2.2.1 Srovnání Z ukázky je zřejmé, že jedním z nejlepe čitelných fontů bude Arial a nebo Verdana za předpokladu, že nebudou řádky v aplikaci příliš dlouhé. Jakmile totiž zvolíme delší text, je na místě patkové písmo.



Obrázek 5: Webové stránky http://www.geocities.com/Vienna/Studio/1541/beethoven.htm



Obrázek 6: Webové stránky http://www.prvni-pomoc.com

Times New Roman
Sylfaen
Georgia
Arial
Verdana

Obrázek 7: Ukázka stylů písma

2.2.3 Volba pozadí a používání barev

2.2.3.1 Co bereme v úvahu při volbě barev:

1. Fyziologické možnosti lidského zraku
2. Psychologické, estetické, kulturní aspekty, tradice ...

2.2.3.2 Jak volit barvy:

1. S barvami bychom měli šetřit, tzn. používat jich co nejméně. Například existují barevná schémata definující jednoduchá pravidla.
2. Poučme se od jiných (vezměme si příklad z přírody, z jiných aplikací...).

2.2.3.3 Srovnání

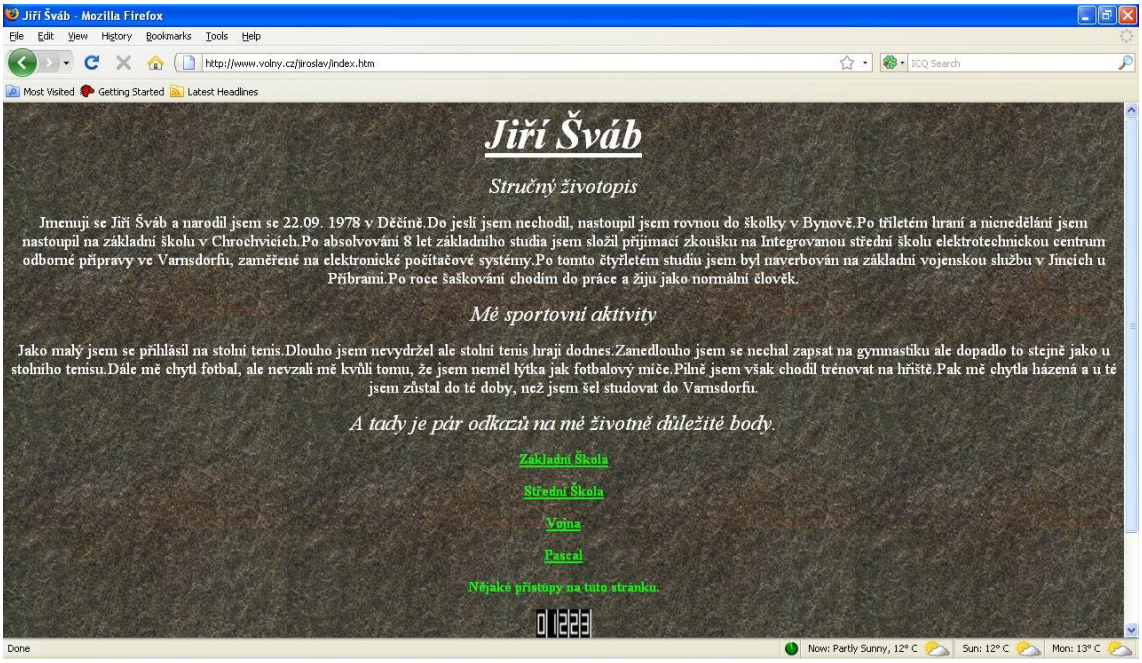
1. Jako příklad špatné volby barev a pozadí jsem vybrala stránku o Jiřím Švábovi, kde se text v zrnitém povrchu pozadí ztrácí. Rušivě zde působí i zelené odkazy. Texty jsou psány patkovým písmem, ačkoliv by se zde mnohem lépe uplatnilo bezpatkové.
2. Na druhé straně obrázek reprezentuje správnou vyváženost barev, kde autor využil teplé odstíny, jež sladil do několika úrovní.

2.3 Nadbytečné grafické prvky

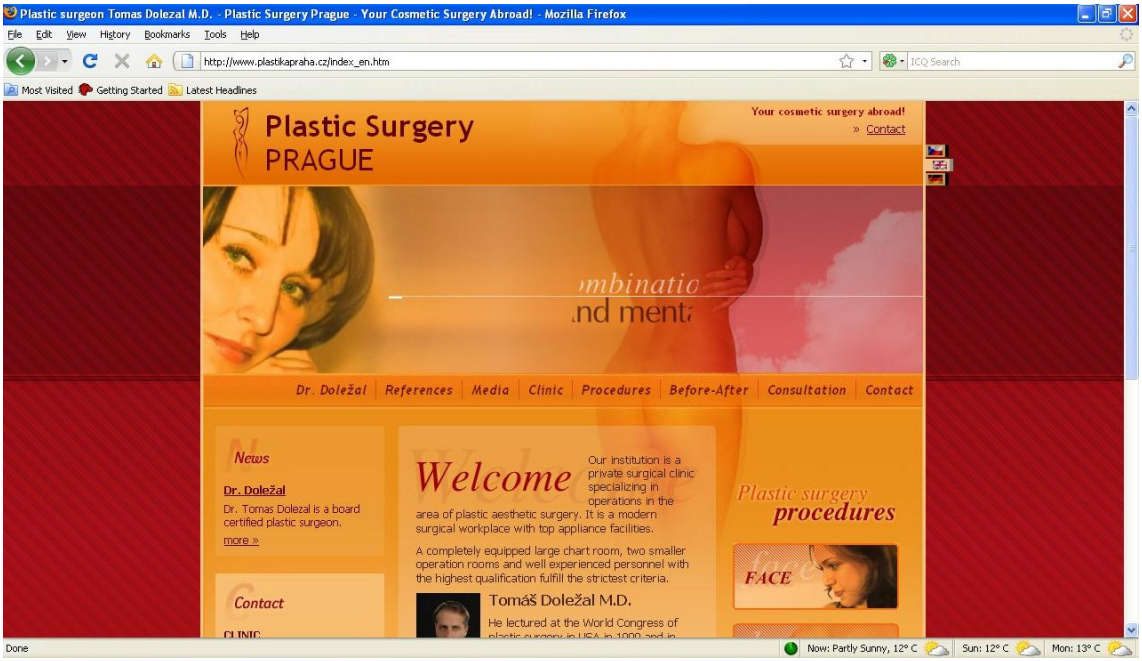
Nadbytečnými grafickými prvky rozumíme ty prvky, které na sebe zbytečně poutají uživatelovu pozornost. Může se jednat například o animace, nevhodně umístěné obrázky, ale také o nic neříkající nadpisy.

2.3.1 Srovnání

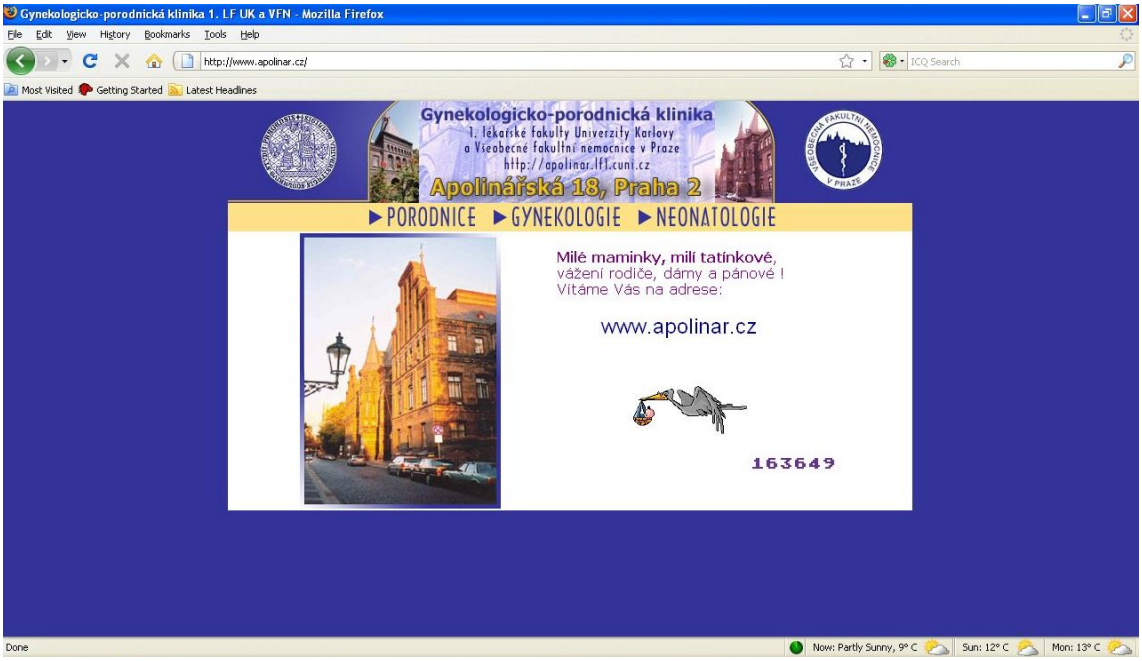
1. Všimněme si obrázku 10, kde vás na první pohled doslova udeří do očí animace čápa s dítkem v zobáku. Teprve poté jste schopni vnímat ostatní věci, což je určitě špatně. Navíc informační stránka webu ustupuje silně do pozadí.



Obrázek 8: Webová stránka http://www.volny.cz/jiroslav



Obrázek 9: Webová stránka http://www.plastikapraha.cz



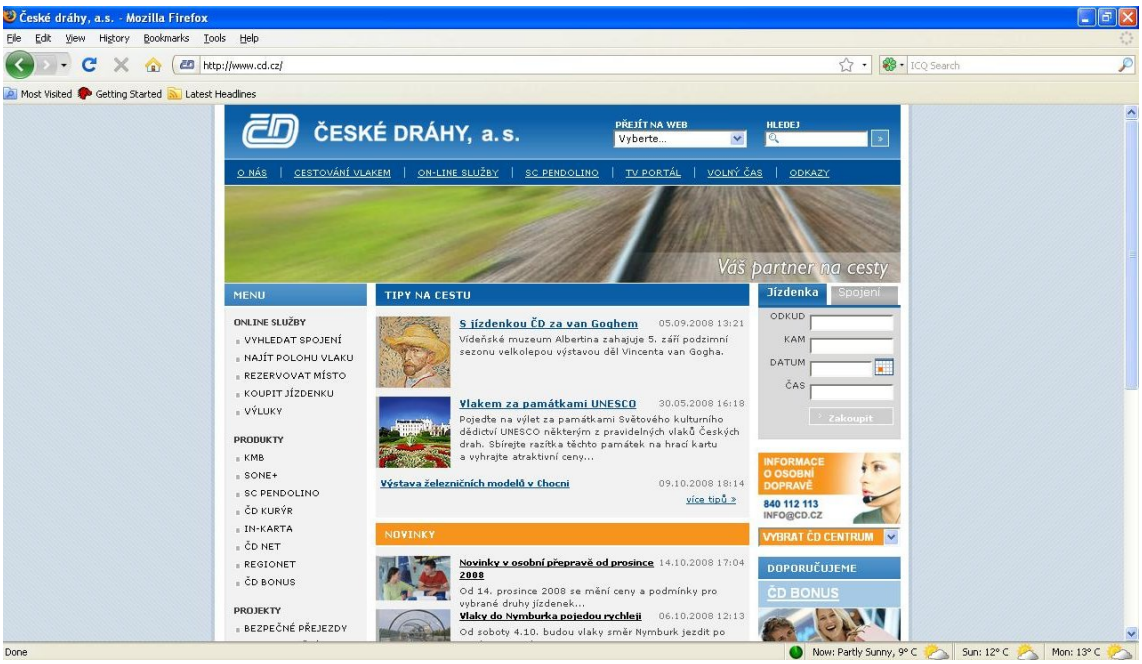
Obrázek 10: Webová stránka http://www.apolinar.cz/

2. Obrázek 11, na němž vidíme web Českých drah a.s. už vypovídá o tom, že je zde informativní stránka zachována a obrázek nahoře je pouze ilustrativní. Web je navíc statický, takže uživatelovu pozornost nebude ovlivňovat žádný pohyb.

2.3.2 Pravidla zacházení s grafickými prvky

1. Obrázek či animace nesmí být stavěna nad informativní stránku aplikace.
2. Pokud si nejste jistí, zda se dané obrázky do aplikace hodí, používejte jich co možná nejméně.
3. V dnešní době je kladen důraz na statický web, tzn. animace jsou vnímány spíše rušivě.

Veškerá pravidla uvedená v této stručné příručce jsou obsažena v [1].



Obrázek 11: Webová stránka http://www.cd.cz/

3 Aplikace Diagramy

3.1 Úvod

V mé vlastní práci, kterou jsem pojmenovala Diagramy se zabývám návrhem co možná nejintuitivnějšího uživatelského rozhraní. Snažím se své rozmístění komponent zdůvodnit, popřípadě srovnat je s ostatními dobře známými programy. Vycházím z výše uvedených pravidel, ze zkušeností a snažím se programy uvedené v této práci hodnotit objektivně a s nadhledem.

3.2 Menu

Jednou z prvních komponent, které uživatele zaujmou, je Menu. Proto je nezbytné, aby bylo uživatelsky co nejpřívětivější, věcné, stručné a hlavně názorné. Je zapotřebí uživateli umožnit rychlou komunikaci s programem Z toho vyplývá, že daným položkám přiřazujeme názvy tak, aby okamžitě pochopil, co znamenají, aniž by je před tím vyzkoušel.

3.2.1 Srovnání s programem Dia

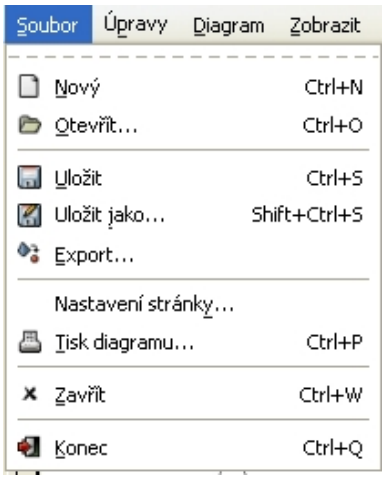
3.2.1.1 Soubor První položkou většiny programů bývá *Soubor*. Protože jsou na to uživatelé zvyklí, nemělo by smysl jejich návyky měnit a nutit je, aby hledali Soubor jinde než právě na začátku.

Standardem jsou rovněž položky menu jako *Nový*, *Otevřít...*, *Uložit* a *Uložit jako...* Následuje Export...Podle mého názoru by tato položka měla být oddělena od ostatních, jelikož nesouvisí ani s uložením ani s načtením stávajících diagramů. Předpokládáme, že uživatelé vědí, co si pod slovem „Export“ představit, nicméně jsem považovala za výhodnější nahradit podstatné jméno „Export“ infitivem „Exportovat“ už jen proto, že předchozí položky nesly v názvu rovněž infinitiv slovesa.

Nastavení stránky... je v menu Soubor naprosto nevhodné, jelikož zde pracujeme se souborem, tedy provádíme souborové akce, nikoliv úpravu výsledného vzhledu stránky.

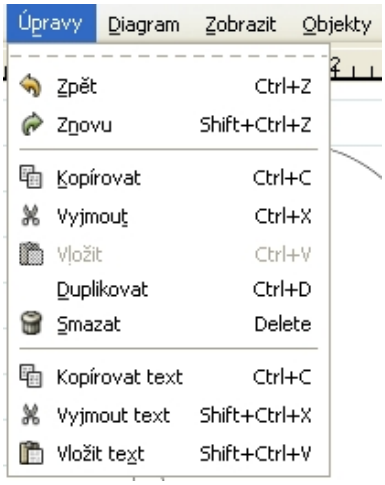
Přestože jsou autoři ve většině případů strozí ve vyjadřování, v případě položky *Tisk diagramu* byly až příliš sdílní. Uživatelé jsou zvyklí z jiných aplikací, at'už se jedná o produkt MS Office nebo tzv. "open source" programy jako například OpenOffice na běžný název *Tisk*. Je to věcné, stručné a naprosto postačující. Je samozřejmostí, že v programu, kde se jedná o diagramy, nebudeme tisknout např. souvislý naformátovaný text, ale právě diagramy.

Naproti tomu položka *Zavřít* je nevyhovující a působí dvojsmyslně. Uživatel by se mohl právem ptát, co chceme zavřít? Celý program nebo pouze diagram, což je funkce této položky? Proto jsem zvolila jednoznačný název *Zavřít diagram*, čímž pochybnosti uživatele zmizí.



Obrázek 12: Program Dia - Soubor

Obrázek 13: Program Diagramy - Soubor



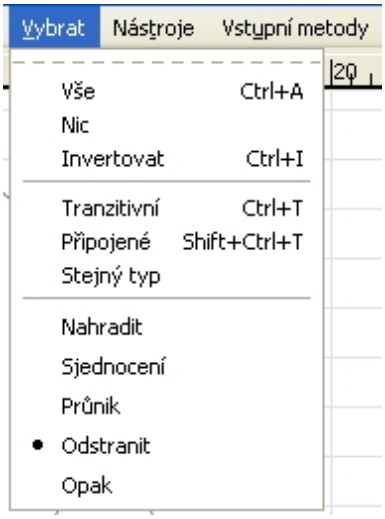
Obrázek 14: Program Dia - Úpravy

Obrázek 15: Program Diagramy - Úpravy

Jakmile bude chtít uživatel aplikaci ukončit, je zvyklý z administrativních programů spíše než na slůvko *Konec* na slovíčko *Ukončit*. Považuji to za zdvořilejší a jemnější vyjádření, což je samozřejmě detail a záleží na názoru každého uživatele, čemu by dal přednost. K přesnějšímu popisu slouží obrázky 12 a 13

3.2.1.2 Úpravy Položka *Úpravy* by měla uživateli sloužit k tomu, aby se mohl vrátit kdykoliv zpět nebo obnovit to, co již vráceno bylo. Uživatel předpokládá, že mu také bude umožněno kopírovat, odebírat, mazat a vkládat zkopírované. To vše je za léta existence grafických rozhraní jakýmsi dogma, jímž se většina programátorů řídí. Porušit je by znamenalo zmást uživatele. Program by tedy ztratil jednu ze základních vlastností, a sice uživatelskou přívětiost.

K položce *Zpět* není co dodat, je jednoznačná a uživatelé jsou na ni zvyklý. Její protipól *Znovu* je diskutabilní. Většina uživatelů ví, co si pod ní představit. Tento název se velmi dobře vžil.



Obrázek 16: Program Dia - Vybrat

Akce *Kopírovat*, *Vyjmout*, *Vložit* a *Smazat* jsou samozřejmostí u každého programu, v němž se nachází objekty popřípadě text či obrázky. Autoři Dia zvolili ještě jednu položku - *Duplikovat*, která je naprosto zbytečná už jen proto, že její funkci plně zastupuje položka *Kopírovat*.

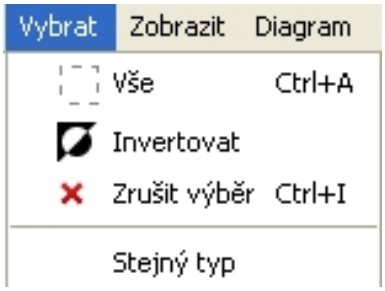
Stejně nesmyslné jsou i *Kopírovat text*, *Vyjmout text* a *Vložit text*. Pro uživatele je totiž lhostejné, zda kopíruje text nebo objekt. Zkrátka potřebuje kopírovat a my – programátoři - bychom mu měli zajistit rychlý přístup k této akci. Tím, že rozlišíme objekty a text, ho dostáváme do pozice, kdy musí při každé této akci přemýšlet, cože to vlastně kopíruje a zda něco nedělá špatně.

K porovnání obou aplikací, tedy Dia a Diagramy slouží obrázky 14 a 15

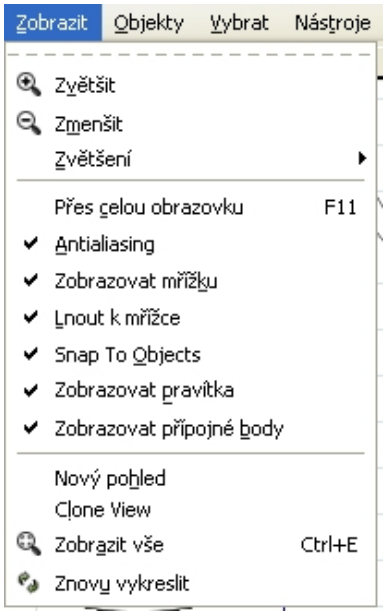
3.2.1.3 Vybrat V programu Dia je tato položka zařazena až daleko za položkou Úpravy, což je z mého pohledu špatně už jen proto, že Úpravy a Vybrat spolu úzce souvisejí.

Zde bych se ráda zastavila až u položky *Nic* (viz. obrázek 16). Ačkoliv by se to mohlo zdát silným označením, nazvu ji absurdní. Jestliže máme položku *Vybrat* a objeví se nám zde položka *Nic*, mateme uživatele, protože pokud nechceme vybrat nic, pak nemusíme klikat na *Vybrat*. Zvolila jsem raději označení *Zrušit výběr* (viz. obrázek 17), kterou uživateli jasně říkám, co se bude dít.

Nahradit, *Sjednocení*, *Průnik* jsou v položce *Vybrat* neprosto nesmyslně pojmenované. Uživatel se může právem ptát: Nahradit – Co čím? Sjednocení – Vybrat sjednocení nebo



Obrázek 17: Program Diagramy - Vybrat



Obrázek 18: Program Dia - Zobrazit

provést sjednocení? Totéž průnik.

Odstranit – jednak jde o úpravu a s výběrem opět nemá nic společného, jednak je tato akce nahrazena položkou *Smazat* v *Úpravách*.

Opravdu bizardní je pak poslední akce *Opak*. Jestliže vybereme nějaký objekt, jaký je jeho opakem? A v případě, že se myslí akce *Invertovat*, pak je naprosto zbytečná a bezpředmětná.

3.2.1.4 Zobrazit Další akce, které se týkají objektů, jsou shrnuty pod položku *Zobrazit*. Jak v aplikaci Dia, tak v mé práci je zařazena na čtvrtém místě.

Obrázek 19: Program Diagramy - Zobrazit

Jako nejvhodnější položkou pro začátek mi připadá *Vše*. Je to dáno tím, že jde o jasnou akci, kterou v případě potřeby uživatel najde okamžitě a ten, jenž ji potřebovat nebude, ji přejde bez většího povšimnutí, protože jde o kratičký text.

Uživatelé jsou z jiných programů zvyklí na to, že v položce Zobrazit naleznou akce jako *Zvětšit* a *Zmenšit*. Ovšem u položky *Zvětšení* v programu Dia je název nevhodný, protože zde nejde pouze o zvětšení, ale také o zmenšování na základě procent. Logicky je tudíž název nesprávný a byl mnou nahrazen položkou *Změnit velikost*.

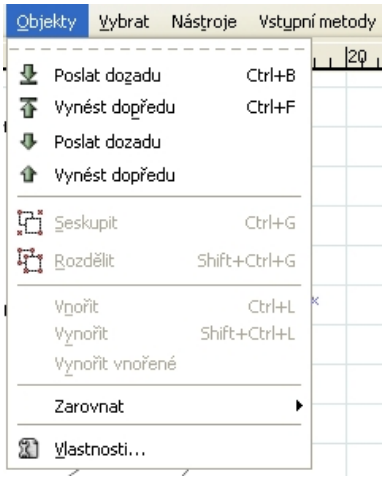
Přes celou obrazovku je zbytečná položka, jelikož každý uživatel zná v pravém horním rohu formuláře skupinu tří tlačítek a může si okno roztáhnout podle potřeby. V Dia programu je to obtížnější tím, že používá dva rozdílné formuláře, které jsou z hlediska ovládání obtěžující. Aplikace Dia navíc při této akci skryje i záhlaví okna a menu, což není potřeba, jelikož monitory jsou v dnešní době dostatečně velké.

Dalším problémem v Dia programu je pak míchání českých a anglických výrazů, což je v naprostém rozporu s tvorbou uživatelského rozhraní i programu jako takového. Většina uživatelů neví, co si má pod položkou Antialiasing nebo Snap To Object představit. A my jej nemůžeme nutit, aby si vyhledával slovíčka ve slovníku.

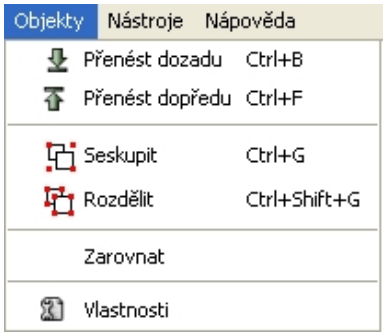
Zobrazit mřížku – nač opakovat *Zobrazit*, když už bylo použito v názvu rodičovské položky?

Lnout k mřížce je diskutabilní. Je to sice způsob, jakým se budou objekty zobrazovat, ale uživatel jej může chápat stejně tak jako způsob chování objektů.

Zobrazovat pravítka, *Zobrazovat přípojné body* – zde nastává stejná situace jako u položky *Zobrazit mřížku*.



Obrázek 20: Program Dia - Objekty



Obrázek 21: Program Diagramy - Objekty

Nový pohled a *Clone View* nejenomže jde o shodné akce, ale navíc se zde opět plete čeština s angličtinou. Nahradila jsem tyto akce jedinou, a sice *V novém okně*.

Znovu vykreslit – zbytečná položka, která může uživatele mást. Nepotřebuje totiž vykreslovat znova své objekty, jelikož program je tak dokonalý, že se zde neobjevují žádné vady již při prvním nakreslení. Překreslování tudíž není nutné.

3.2.1.5 Objekty

Dostáváme se k manipulaci se samotnými objekty.

Poslat dozadu je z mého pohledu poněkud irelevantní označení této akce, proto jsem nahradila slůvko „poslat“ slůvkem „přenést“, jelikož zní zdvořileji. *Vynést dopředu* je rovněž nepřijatelný výraz. Raději *Přenést dopředu*, na což jsou uživatelé zvyklí z jiných aplikací.

K dalším dvěma položkám *Seskupit* a *Rozdělit* není zapotřebí komentáře. Objevují se ve většině programů a uživatelé je hojně využívají.

Podívejme se nyní na položky *Vnořit*, *Vynořit*, *Vynořit vnořené*. Akce, jež ani svým názvem nenapovídají, o co by mělo jít. Co se bude kam vnořovat nebo odkud vynořovat. Zkrátka položky zavádějící, které nutí uživatele zkoumat, o jakou operaci půjde. Zdržují a odvádějí od práce. Ovšem na druhou stranu, proč bychom tyto komponenty neumístili do panelu, jež se zobrazí po stisknutí pravého tlačítka na objekt? V takovém případě uživatel porozumí kontextu a pochopí, co mu panel nabízí.

Zarovnat je samozřejmostí, aby diagram vypadal upraveně a byl hoden prezentovat nějaký problém. Proto uživateli tuto akci umožníme.

Vlastnosti jsou zde na správném místě. Jde totiž o vlastnosti objektů, kde si můžeme například zvolit tloušťku čáry nebo barvu.

Položku *Objekty* reprezentují obrázky 20 a 21, kde jsou změny dobře patrné.

3.3 Panel nástrojů – přímka

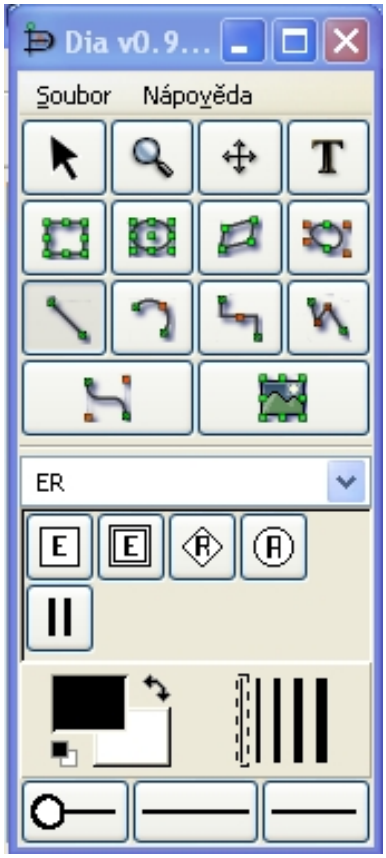
Přímka je dle mého názoru jeden z nejdůležitějších prvků *Panelu nástrojů*, jelikož její využití se prolíná napříč všemi možnými diagramy. Je proto důležité, aby byl co možná nejpřehlednější a navržen podle standardů, na něž jsou uživatelé zvyklí.

3.3.1 Umístění

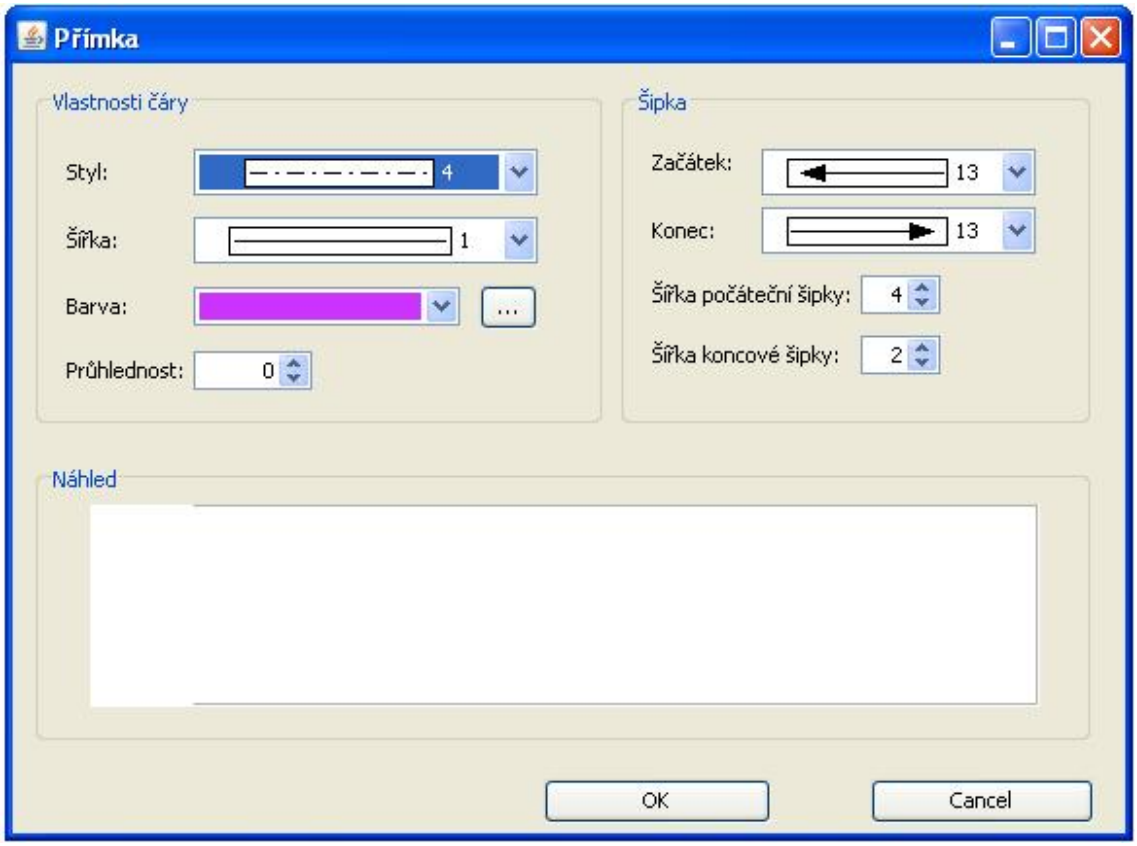
Ze všeho nejdříve je důležité si rozmyslet, kam vlastně tento panel a odkaz na něj umístíme. Když se podíváme na program Dia, vidíme, že v podstatě všechny vlastnosti přímky jsou umístěny v levém panelu nástrojů. To může být za daných okolností velice pohodlné. Uživatel má vše jako na dlani a nemusí se probírat složitými menu. Ovšem na druhé straně Dia má dva panely. Jeden – Panel nástrojů a druhý, kam můžeme tvořit diagramy. Dva panely jsou z mého pohledu velice nešikovné, jelikož zabírají místo nejen ve startovní liště, ale také znemožňují uživateli pohodlnou práci na celé obrazovce. Jestliže uživateli nedáme možnost, aby si zvolil, na jak velké ploše chce pracovat a omezíme ho, je velká pravděpodobnost, že bude hledat lepší nástroje, které jej omezovat nebudou.

3.3.2 Rozmístění komponent

Je všeobecně zažito, že v programech, kde jsou panely nástrojů s položkou Přímka, začínají s rozmístěním takto: Nejprve v levém horním rohu umisťují panel s Vlastnostmi čáry, což je pochopitelné, jelikož se předpokládá, že uživatele v tomto panelu budou zajímat právě šířka čáry, její styl a barva. Vedle pak tvůrci uživatelských rozhraní pokládají panel pro nastavenní šipek. Je nápadné, že málokdo umisťuje tento panel pod panel s Vlastnostmi čáry. Většinou je tento umístěn vpravo, právě vedle Vlastností čáry. To



Obrázek 22: Program Dia - Přímka



Obrázek 23: Program Diagramy - Nástroje - Přímka

může mít hned několik důvodů. Jednak ušetříme místo, protože kdybychom vložili tento panel pod Vlastnosti čáry, celý panel by se neúměrně protáhl a uživateli by se mohl zdát příliš složitým. Dalším důvodem může být umístění Náhledu. Náhled bývá zpravidla uložen ve spodní části celého panelu, je to jakési shrnutí všeho nastavení, jaké uživatel provedl. Z tohoto důvodu by měl být dostatečně velký a zřetelný. Jestliže bychom již zmíněné komponenty umístili pod sebe, panel s náhledem by se zúžil. Nehledě na to, že naším úkolem je uživateli umožnit intuitivní práci, kde nemusí nic příliš hledat, a toho docílíme jedině tak, vezmeme - li si příklad z aplikací, které denně při práci používá. Např. Microsoft Word, OpenOffice a jiné, které dávno určily, jak by mělo rozhraní pro tvorbu přímky vypadat.

3.3.3 Komponenty

Z mého hlediska by mělo být co možná nejvíce komponent grafických. Obrázkům totiž porozumí většina , je - li dobře nakreslen a ukazuje - li danou funkci. Proto volíme grafické vyjádření i stylu čáry, její barvy a šířky. Pro uživatele je to pohodlnější a smyslově rychlejší. Protože mu bude trvat kratší dobu, než pochopí obrázek, narozdíl od čtení textu.

Otázkou zde zůstává, jak docílit grafického vyjádření a kterou komponentu zvolit. Mnohé programy využívají k volbě např. barvy čáry obyčejný Combobox, tedy komponentu, kde se dá zvolit ze seznamu, kde nadefinují, samozřejmě s patřičnými ukázkami odstínů, příslušné barvy. Jiné dávají přednost tzv. Color Chosseru, z něhož si uživatel může vybrat okamžitě, jelikož vidí všechny barvy vedle sebe najednou.

Z mého pohledu je nejlepším řešením právě již zmíněný Color Chooser, ovšem s nabídkou dodefinování vlastních barev, abychom se vyhnuli tomu, že uživatel bude v budoucnu nějakou z barev postrádat. Skvělou možností, jak dát uživateli co možná největší prostor a zajistit jeho oblíbenou volbu jsou záložky, kde si smí zvolit z RGB, HSV a tradiční tzv. Swatches. Tím je zabezpečeno pohodlné použití pro široké spektrum uživatelů. Ne každý totiž holduje RGB modelu a ne každý je zastáncem omezeného spektra paletky Swatches. K tomu, abychom zajistili okamžitou odezvu a uživatel ihned viděl, jak daná barva bude vypadat na jeho přímce, je dle mého názoru nejvhodnější položkou ComboBox, který se obarví podle zvolené barvy. Jednak je to názorné a interaktivní a jednak jsou na to uživatelé zvyklí z jiných aplikací. V uživatelských rozhraních více než jinde platí, že zvyk je železná košile. Proto bychom se toho měli držet.

Jakmile definujeme šipky, je to obdobné. Nejlepší jsou grafická znázornění ukončení šipek, která dají uživateli jasně najevo, jak bude daná komponenta po vytvoření vypadat.

Pokud jde o SpinBoxy, tedy komponenty, kde si šipkami zvyšujeme nebo snižujeme číselnou hodnotu, jež nám ukazují procenta a jiná čísla, považuji je za nejlepší možnost v kombinaci s náhledem. Jesliže umožníme uživateli náhled ihned po zvolení, pak je velmi pravděpodobné, že si zapamatuje hodnotu, jakou potřeboval a příště už ji automaticky pro ještě rychlejší práci napíše do spinboxu sám, aniž by hledal příslušnou hodnotu v

náhledu.

Mnozí mohou namítat, že posuvníky jsou ještě lepším řešením. Mohou být v případě, že poblíž nich se bude ukazovat hodnota, jakou uživatel volí. Pro běžného uživatele sice procentuální vyjádření sytosti nemusí být důležité, nicméně neměli bychom podceňovat uživatelské schopnosti a vědomosti a dát jim možnost vidět hodnotu zvolenou pro svou přímku.Čímž vyvstává problém právě u posuvníků, u nichž musíme nutně přidat další komponentu, a sice Label, aby byla hodnota patrná. Zatímco u Spinboxů je hodnota patrná, volení je jednoduché a detailní zobrazení zaručí náhled. To máme o celou jednu komponentu méně. Ušetříme místo a zajistíme lepší přehlednost.

Všechny tyto vlastnosti obsahuje program Diagramy (viz. obrázek 23)

3.4 Srovnání s ostatními programy

Aby bylo ještě lépe patrné, co se v dnešních programech používá a na co si uživatelé byli nuceni zvykat, popřípadě, na co už si zvykli, přináším zde několik ukázek z grafických prostředí. Jedná se o programy bez ohledu na to, zda jsou určeny ke kreslení diagramu nebo k něčemu zcela jinému. Nejdůležitějším měřítkem je zde panel nástrojů pro definici přímky.

3.4.1 Open Office Writer

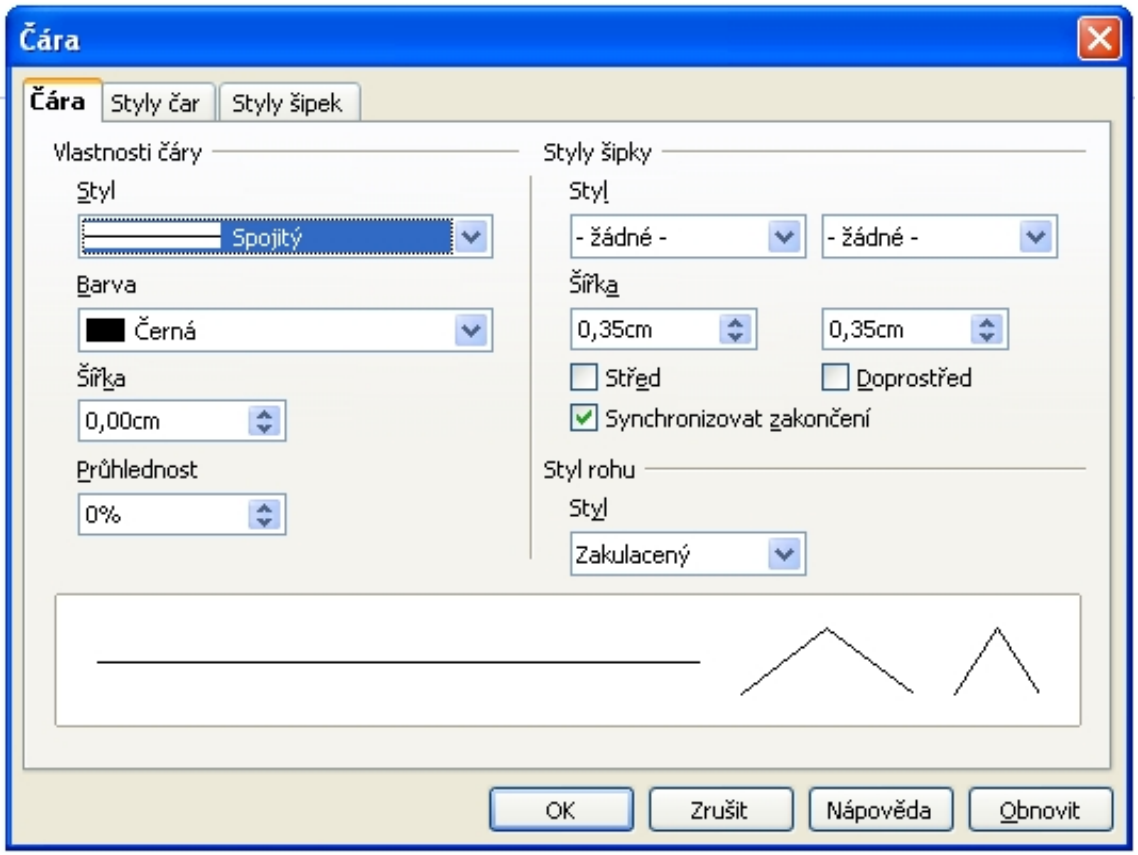
Jako první uvádím ukázku z OpenOffice Writeru (na obrázku 24), jenž podobně jako Microsoft Word umožňuje definovat jednoduché obrazce a tvary, mezi nimi i přímku.

Když se na obrázek zadíváme pozorněji, vidíme, že panel obsahuje tři záložky. Podle mého názoru je to zbytečně moc. Všechny položky se dají shrnout do jedné, aniž bychom způsobili nepřehlednost nebo snad chaos. Spousta věcí se v daném panelu opakuje, což může vést k matení uživatele.

Všimneme – li si navíc ještě ohraničujících čar, jež od sebe oddělují například Vlastnosti čáry a Styly šipky, dospějeme nutně k názoru, že ohraničující čáry zde působí neuceleně a velmi rušivě, jelikož na sebe strhávají pozornost tím, že jsou neucelené.

3.4.2 Microsoft Visio

Pokud Microsoft Visio (ukázka na obrázku 25) srovnáme s předchozí aplikací, vidíme, že začátek je velice podobný, ne – li stejný. Za pozornost zde stojí posuvník s názvem Transparency. Vedle něj je pak textové pole, kde je možno vidět, jakou průhlednost nastavujeme. Tedy jde o další komponentu, jak jsem již zmínila na začátku kapitoly. Z estetického hlediska navíc posuvník působí poněkud rušivě.



Obrázek 24: Program OpenOffice Writer - Panel nástrojů - Přímka

Na rozdíl od Open Office Writeru je zde přidána velikost šipky na začátku a na konci, což je pro uživatele výhodné, nelíbí – li se mu běžný předdefinovaný styl. Z druhého úhlu pohledu však může být i ke škodě, protože např. technická šipka má přesně stanovenou velikost.

Možnost nastavení ohybu přímky je zde něco, co v předchozích aplikacích většinou chybí. Ohyb přímky je specialitou Visia. Čára se totiž může ohýbat nebo zvlnit. Pro uživatele, jenž neumí s takovouto čarou pracovat a je zvyklý, že čára bývá zpravidla rovná, může práce s takovými možnostmi představovat jistá úskalí. Nehledě na to, že například v ER diagramech nebývají většinou takovéto čáry zapotřebí a preferují se skutečně rovné přímky.

Zde je náhled umístěn v pravém horním rohu. Je s podivem, že právě Microsoft se rozhodl k takto radikálnímu řezu, jelikož jeho populární MS Word používá náhledu vždy v dolní části panelu. Ačkoliv jde jen o detail, z estetického hlediska a hlavně z hlediska uživatele jde o nešikovné řešení, jelikož uživateli bere již vžité umístění.

Tlačítko Apply zde plní funkci Ok, nevidím tedy důvod, proč by měl uživatel volit mezi těmito dvěma tlačítky a proč jej program nutí přemýšlet, jaký je mezi nimi rozdíl a co se stane, když místo Apply klikne na Ok a naopak.

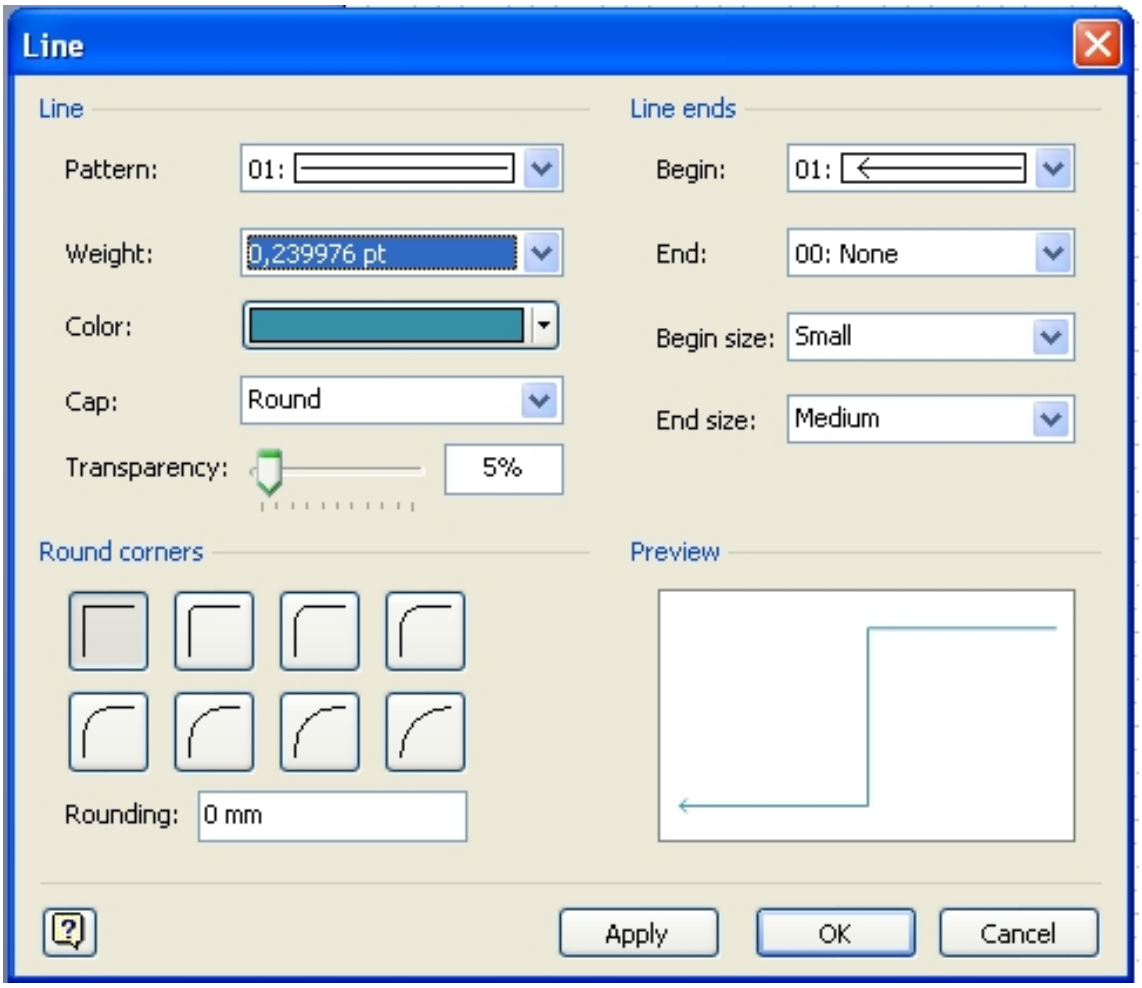
3.4.3 ArchiCAD

ArchiCAD (obrázek 26) zvolil velice jednoduchý nástroj, kde ovšem uživatel velice rafinovaným způsobem najde vše, co potřebuje. Sekce jsou od sebe zřetelně odděleny, což budí dojem spořádanosti.

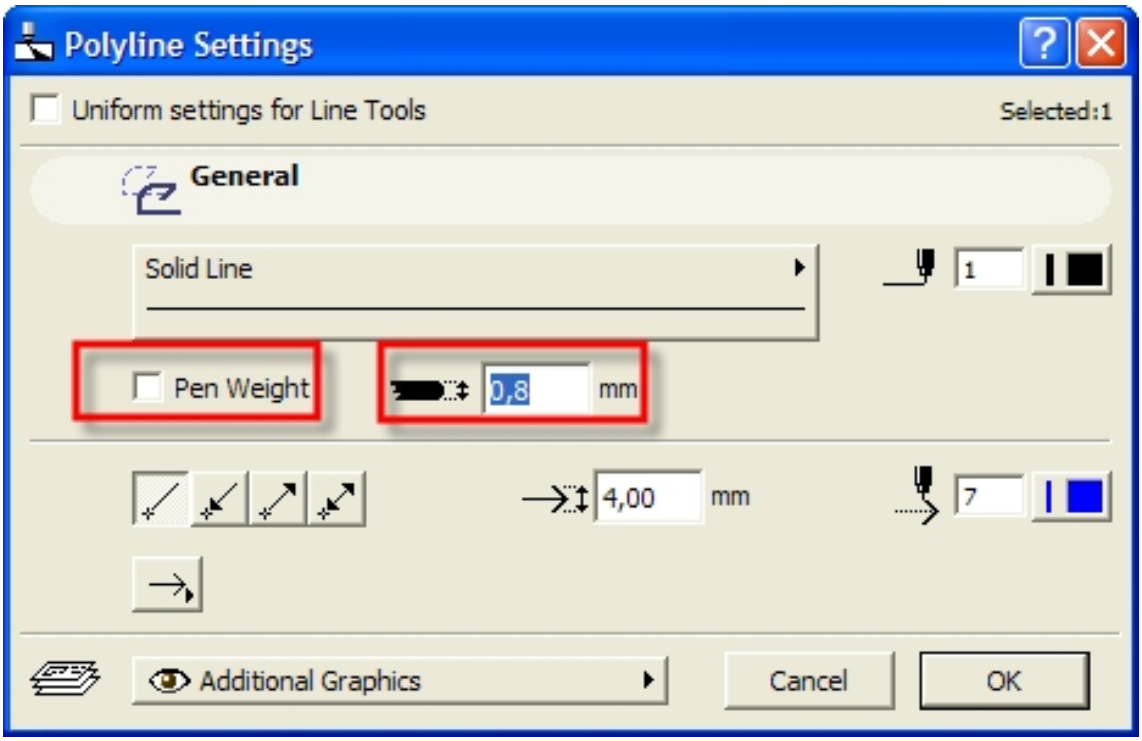
Co zde může působit poněkud rušivě a zbytečně poutat pozornost je lišta s nápisem General a drobný obrázek. Je to totiž v dané sekci nejvýraznější položka.

Uživatel si zde může zvolit přednastavené typy čáry, a nebo jednoduše nadefinovat svou čáru o své šířce, což je velice praktické pro profesionály, o nichž se předopokládá, že budou s programem pracovat.

ArchiCAD se držel co nejvíce tlačítek. Je patrné, že se vyhnul komponentám typu ComboBox nebo SpinBox, raději zvolil tlačítka, jež se dají rozkliknout na další z nabídek. Tím samozřejmě šetří místo. Ovšem na druhé straně neumožnil uživateli náhled, což je podle mého názoru v programu, který je ryze grafický poměrně velká chyba.



Obrázek 25: Program MS Visio - Panel nástrojů - Přímka



Obrázek 26: Program ArchiCAD - Panel nástrojů - Přímka

3.5 Grafické objekty

Program Diagramy nemá za úkol zvláštní funkčnost, nesnažíme se zde ukázat, jak by měla aplikace fungovat, ale zabýváme se zde uživatelským rozhraním a vlivem stylu uživatelského rozhraní na rychlost a orientaci uživatele.Zde jsme pro ilustraci zprovoznily pouze několik komponent, mezi něž patří Obdelník, Ovál a Přímka.

3.5.1 Srovnání s jinými uživatelskými rozhraními

Jestliže vedle sebe postavíme například obyčejné Malování ve Windows, Dia a Visio, uvidíme hned několik způsobů umístění objektů na plochu. Každé má své výhody a nevýhody. Z uživatelského hlediska proto nemůžeme zcela určitě říci, co je správné a co je naprosto špatně, jelikož uživatel je bytost jedinečná a má rovněž jedinečný náhled. Dále je třeba vzít v úvahu i to, že návrh aplikace se často liší podle cílové uživatelské skupiny.

3.5.1.1 Malování ve Windows V Malování si uživatel v levém panelu vybere komponentu a klikne do malovací plochy, kde se teprve po kliknutí začíná objevovat objekt. To je z uživatelského hlediska velice příznivé, jelikož odpovídá reálnému modelu. Člověk, pokud hodlá malovat, nejprve vezme do ruky tužku a teprve poté, co se dotkne papíru jejím hrotem, začíná reakce, a tou je právě nakreslení obrázku. Příznivě zde také vyznívá změna kurzoru v tužku, takže uživatel ví, co se bude dít a co právě provádí.

3.5.1.2 Dia Dia v podstatě jde stejnou cestou. V pravém okně si vybereme komponentu, jež má přednadstavené jisté vlastnosti a teprve po kliknutí do kreslící plochy se kreslí objekt.

3.5.1.3 Microsoft Office Visio Microsoft Office Visio používá odlišný náhled na věc, a sice používá pro uživatele model, jenž se velice podobá situaci, kdy něco stavíme. Představme si například stavění domečku. Vezmeme kostku a přetáhneme ji na námi požadované místo. Totéž se děje i s objektem v aplikaci. Uživatel stiskne na položce levé tlačítko myši a pak daný objekt přemístí na kreslící plochu. Dalo by se říci, že tzv. přetahování objektu myší je atomická operace a uživatele nenutí si pamatovat, který objekt si zvolil.

3.5.1.4 Open Office Writer Podívejme se na programy, které nejsou přímo určeny k výrobě diagramů. Open Office Writer má rovněž předdefinované tvary proto, aby zde mohl být nějakým způsobem vytvořen diagram. Tedy zvolíme si objekt a vidíme okamžitou reakci aplikace, protože se nám kurzor mění v osový kříž, jímž budeme kreslit objekt. To je pro uživatele jasná zpětná vazba, o níž jsem mluvila na začátku. Zpětná vazba je totiž důležitá. Tak, jako v dětství poznáváme svět, tak i nyní, v dospělosti, je pro nás tato vazba velmi důležitá.

3.5.1.5 Gimp Gimp nabízí totéž, co zmíněné malování a ostatní programy, tedy zvolení objektu, změnu kurzoru a následné kreslení.

3.5.2 Vyvozený závěr

Ačkoliv jsem na začátku uvedla, že nelze říci, co je zde dobře a co špatně naprosto nekompromisně, mohu s klidným svědomím prohlásit, že pokládám za logičtější ty programy, jež využívají jednak zpětnou vazbu a jednak zaběhnutého režimu uživatele. V případě, že vycházíme z reality a držíme se jejího obrazu, pak bychom neměli učinit nikde chybu, jelikož je to právě realita, nabyté zkušenosti a vědomosti, které nás učí, jak které věci provádět. Z aplikací, které jsem zde jmenovala, jsme poznali hned několik způsobů řešení daného problému. A ani jeden nelze odsoudit jako naprosto špatný.

3.5.3 Řešení objektů v BP

Po prostudování výše zmíněných faktů, jsem i já začala uvažovat, jak vyřešit tuto situaci. Nakonec jsem se rozhodla, že budu vycházet z aplikací, které se drží uživatelovi reality a zpětnou vazbu vyřeším po svém.
V aplikaci Diagramy uživatel stiskne tlačítko – tedy zvolí si komponentu a komponenta se mu ihned umístí na plochu. Má pochopitelně předdefinované vlastnosti, to ovšem nepokládám za chybu, jelikož uživatel, jak vysvětlím později, si může komponentu sám doupravit.
V čem spatřuji klad mého řešení: Zpětnou vazbu jsem neřešila změnou kurzoru, objekt je okamžitě umístěn na plochu a uživateli stačí vzít jej, přetáhnout ho tam, kam potřebuje a libovolně si editovat vlastnosti. Navíc zvolený objekt je již předdefinovaný – tedy má určitou výšku a šířku, a uživatel si jej později přizpůsobí k obrazu svému.

3.5.4 Editace objektů

Pochopitelně, že v každém uživatelském rozhraní, které se zabývá návrhem diagramů, je nutno umožnit uživateli editaci objektu. Máme totiž několik možností, jak to provést. Spousta programů používá pro editaci tzv. čtverec – jde o malou oblast kolem objektů, s nímž lze pohybovat a díky němuž se vlastnosti objektů - tedy délka a šířka – mění. Další možností jsou tzv. editační body.
V mém případě došlo právě na zmíněné editační body. Uživatelé jsou na ně zvyklí, navíc já zde srovnávám svou aplikaci s aplikací Dia a ne vše, jak by se mohlo zdát z mé práce, v ní kritizuji. Snažila jsem se vyjít právě z ní, kde se objekty editují takto. Proto je kolem nich umístěn různý počet editačních čtverečků. Aby byly lépe patrné a uživatel rychleji objevil, že se něco děje, nechala jsem je obarvit na zeleno, jako právě v Dia.
Za tyto čtverečky může uživatel tahat, čímž se bude měnit velikost a do jisté míry i tvar daného objektu. Je pochopitelné, že počet těchto čtverečků jsem volila dle zvyklosti, proto má např. obdelník a ovál čtverečky čtyři, zatímco přímka pouze dva.
Uživatel má samozřejmě právo po ploše objekty rozmisťovat dle svého uvážení a přání,

proto jsem i já umožnila posouvání objektů. Děje se tak, jakmile uživatel uchopí objekt – kliknutím dovnitř objektu. Přímka je řešena samozřejmě jinak, tam nelze uživatele nutit, aby klikal přesně na přímku, jež je velice tenká. Proto jsem kolem přímky umístila neviditelné pole, jež toleruje určitou vzdálenost kliknutí na přímku. Díky tomu lze tedy přímku posouvat rovněž.

Pro tvorbu samotné aplikace byl klíčovým studijním materiálem [1, 2].

4 Závěr

Má aplikace Diagramy slouží jako ukázka toho, jak vytvořit uživatelské rozhraní podle definovaných zásad. Nelze ji tedy považovat za plnohodnotnou. Neměla jsem za úkol zabývat se funkčností, ačkoliv částečně v ní zahrnuta je. Snažila jsem se využít všechny prostředky, abych dosáhla modelu, který by mohl být vzorem pro ostatní aplikace. Žádný člověk však není neomylný a říká se: Sto lidí, sto názorů. Ačkoliv jsem omezila své názory na minimum, nelze je zcela z práce vyloučit, neboť každý, kdo studuje určité téma, si na něj vytvoří osobní pohled.

5 Literatura

[1] Sojka, Eduard, přednášky z předmětu *Uživatelská rozhraní*, 2004/2005.

[2] Neměc, Martin, přednášky z předmětu *Základy počítačové grafiky*, 2008/2009.

[3] Brůha, Lubomír, *Java Hotová řešení*, Brno: Computer Press, 2003.